Source code :

```
#Confusion Matrix
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv("ds_guarding_customer_churn_500.csv")
#Print(df)
#print(df.head())

cat_cols = ['gender', 'contract_type', 'internet_service', 'payment_method']
le = LabelEncoder()
for col in cat_cols:
    df[col] = le.fit_transform(df[col])

X = df.drop(['customer_id', 'churn'], axis=1)
y = df['churn']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

#Line plot
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv("ds_guarding_customer_churn_500.csv")
#Print(df)
#print(df.head())

for col in df.select_dtypes(include='object').columns:
    if col != 'customer_id':
        df[col] = LabelEncoder().fit_transform(df[col])

df_corr = df.drop(columns=['customer_id'])
```

```
corr_matrix = df_corr.corr()

#Line plot
plt.figure(figsize=(12, 6))
plt.plot(np.arange(len(y_prob)), y_prob, label='Churn Probability', color='red')
plt.axhline(y=0.5, color='black', linestyle='--', label='Decision Threshold')
plt.title("Line Plot of Predicted Churn Probabilities")
plt.xlabel("Customer Index")
plt.ylabel("Predicted Churn Probability")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

y_pred = model.predict(X_test)
print("Classification Report:\n", classification_report(y_test, y_pred))

cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['No Churn',
'Churn'], yticklabels=['No Churn', 'Churn'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

#Boxplot
import matplotlib.pyplot as plt
import seaborn as sns
from IPython import get_ipython
from IPython.display import display
import pandas as pd
import numpy as np

data = {
    'numeric_feature1': np.random.rand(100),
    'numeric_feature2': np.random.randint(1, 100, 100),
    'categorical_feature': ['A', 'B'] * 50,
    'churn': np.random.randint(0, 2, 100)
}
df = pd.DataFrame(data)
```

```
numeric_features = df.select_dtypes(include=[np.number]).columns.tolist()

if 'churn' in numeric_features:
    numeric_features.remove('churn')

if len(numeric_features) == 0:
    print("No numeric features found to plot.")
else:
    n_features = len(numeric_features)
    n_cols = 2
    n_rows = (n_features + n_cols - 1) // n_cols

    plt.figure(figsize=(14, n_rows * 5))

    max_plots = 4
    features_to_plot = numeric_features[:max_plots]

    for i, feature in enumerate(features_to_plot, 1):
        plt.subplot(n_rows, n_cols, i)
        sns.boxplot(data=df, x='churn', y=feature, palette='Set2')
        plt.title(f'Boxplot of {feature} by Churn')
        plt.xlabel("Churn (0 = No, 1 = Yes)")
        plt.ylabel(feature)

    plt.tight_layout()
    plt.show()

#KTE Diagram
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder, StandardScaler


df = pd.read_csv("ds_guarding_customer_churn_500.csv")
#Print(df)
#print(df.head())

for col in df.select_dtypes(include='object').columns:
    if col != 'customer_id':
        df[col] = LabelEncoder().fit_transform(df[col])
```

```
X = df.drop(['customer_id', 'churn'], axis=1)
y = df['churn']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
model = RandomForestClassifier(random_state=42)
model.fit(X_scaled, y)

importances = model.feature_importances_
feature_names = X.columns
indices = np.argsort(importances)[::-1]

plt.figure(figsize=(12, 6))
sns.barplot(x=importances[indices], y=feature_names[indices], palette='viridis')
plt.title("KTE Diagram: Feature Importances for Customer Churn Prediction")
plt.xlabel("Importance Score")
plt.ylabel("Feature")
plt.tight_layout()
plt.show()

#Heatmap
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv("ds_guarding_customer_churn_500.csv")
#Print(df)
#print(df.head())

for col in df.select_dtypes(include='object').columns:
    if col != 'customer_id':
        df[col] = LabelEncoder().fit_transform(df[col])

df_corr = df.drop(columns=['customer_id'])

corr_matrix = df_corr.corr()

plt.figure(figsize=(14, 10))
sns.heatmap(corr_matrix, annot=True, fmt=".2f", cmap='coolwarm', square=True,
linewidths=0.5)
plt.title("Heatmap of Feature Correlations")
plt.tight_layout()
plt.show()
```